Package 'hotpatchR'

April 24, 2026

Type Package

Title Runtime Namespace Patching Utilities for R Packages

Version 0.1.0

Author David Munoz Tord [aut, cre]

Maintainer David Munoz Tord <david.munoztord@mailbox.org>

Description Provides utilities for runtime hotpatching of locked R package namespaces. The package enables dynamic injection of function patches into sealed package environments without rebuilding or redeploying the package. This is particularly useful for legacy containerized workflows where package versions are frozen in place. The core functionality includes inject_patch() to inject patches into package namespaces, undo_patch() to restore original functions, apply_hotfix_file() to apply patches from external R scripts, and test_patched_dir() to run test suites against patched packages. The package implements namespace surgery techniques that allow internal callers to automatically see patched functions.

License MIT + file LICENSE

Encoding UTF-8

Imports testthat (>= 3.0.0)

Suggests knitr, rmarkdown, pkgdown

VignetteBuilder knitr

RoxygenNote 7.3.2

NeedsCompilation no

Repository CRAN

Date/Publication 2026-04-24 18:20:07 UTC

Contents

`apply_hotfix_file` *Apply a hotfix file and inject the patch definitions*

Description

Apply a hotfix file and inject the patch definitions

Usage

```
apply_hotfix_file(file, pkg = NULL)
```

Arguments

`file`	Path to an R script that defines ‘patch_list‘ and optionally ‘pkg‘.
`pkg`	Optional package name if not provided in the script.

Value

Invisibly TRUE on success.

Examples

```
# Create a temporary hotfix file
hotfix_content <- "
pkg <- 'hotpatchR'
patch_list <- list(
  dummy_child_func = function(x) {
    paste('HOTFIXED! Input:', x)
  }
)
"

hotfix_file <- tempfile(fileext = ".R")
writeLines(hotfix_content, hotfix_file)

# Apply the hotfix from file
apply_hotfix_file(file = hotfix_file, pkg = "hotpatchR")

# Verify the patch works
result <- dummy_parent_func("test")
print(result)

# Clean up
unlink(hotfix_file)
```

dummy_child_func *A dummy child function that we will "hotfix" in tests*

Description

A dummy child function that we will "hotfix" in tests

Usage

```
dummy_child_func(x)
```

Arguments

x	Input value

Value

A character string with child output

Examples

```
# This function is called by dummy_parent_func
# It serves as an example of a function that can be hotpatched
result <- dummy_child_func("example")
print(result)
```

dummy_parent_func *A dummy parent function to test namespace injection*

Description

A dummy parent function to test namespace injection

Usage

```
dummy_parent_func(x)
```

Arguments

x	Input value to pass to child function

Value

A character string with parent output

Examples

```
# Call the parent function which internally calls dummy_child_func
result <- dummy_parent_func("sample")
print(result)
```

`inject_patch` *Inject a runtime patch into a locked package namespace*

Description

Inject a runtime patch into a locked package namespace

Usage

```
inject_patch(pkg, patch_list, lock = TRUE)
```

Arguments

`pkg`	A package name as a string.
`patch_list`	A named list of functions to overwrite in the package namespace.
`lock`	Whether to re-lock bindings after patching.

Value

Invisibly TRUE on success.

Examples

```
# Show baseline behavior with broken function
baseline <- dummy_parent_func("test")
print(baseline)

# Inject a patched version of the child function
inject_patch(
  pkg = "hotpatchR",
  patch_list = list(dummy_child_func = function(x) {
    paste("I am the FIXED child! Input:", x)
  })
)

# Call the parent function again - it now uses the patched child
patched_result <- dummy_parent_func("test")
print(patched_result)
```

test_patched_dir *Run testthat tests against a patched package namespace*

Description

Run testthat tests against a patched package namespace

Usage

```
test_patched_dir(pkg, test_path = "tests/testthat", reporter = "summary")
```

Arguments

pkg	Package name that has been patched in memory.
test_path	Path to tests to run.
reporter	testthat reporter name or object.

Value

Result object from testthat::test_dir.

Examples

```
## Not run:
  # Inject a patch to the package
  inject_patch(
    pkg = "hotpatchR",
    patch_list = list(dummy_child_func = function(x) {
      paste("PATCHED! Input:", x)
    })
  )

  # Run tests against the patched package
  test_patched_dir(pkg = "hotpatchR")

## End(Not run)
```

undo_patch *Undo a previously injected patch*

Description

Undo a previously injected patch

Usage

```
undo_patch(pkg, names = NULL)
```

Arguments

pkg	Package name or environment.
names	Character vector of patched object names to restore. If NULL, restore all stored backups for pkg.

Value

Invisibly TRUE on success.

Examples

```
# First inject a patch
inject_patch(
  pkg = "hotpatchR",
  patch_list = list(dummy_child_func = function(x) {
    paste("I am PATCHED! Input:", x)
  })
)

# Call with patched function
patched <- dummy_parent_func("test")
print(patched)

# Restore the original function
undo_patch(pkg = "hotpatchR", names = "dummy_child_func")

# Now it's back to the original
restored <- dummy_parent_func("test")
print(restored)
```

Index